HyperLogLog: Analysis and implementation of an improved algorithm

Chloé Dequeker, Ghiles Ziat

13 fevrier 2015

Introduction

Cardinality estimation problem :

- The naive solution does not scale !
- Several alogorithms have been proposed

Today, we'll talk about :

HyperLogLog++ (call it HyperGoogle)

Improvement of the HyperLogLog

Hyperloglog

The approach of the HyperLogLog :

- Randomization using a hash function
- Observation of the maximum of the number of leading zeros
- Stochastic averaging

The result is then subjected to corrections

- Small range correction
- Large range correction

Bias estimation and correction

Transition to 64 bits → an increase of the efficiency area

Bias

The observed bias depends on the cardinality estimated. A correction then can be computed

- Bias estimation
- Store them into a file
- File loading
- Linear interpolation

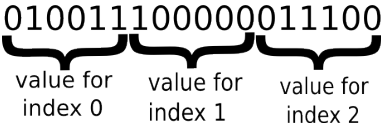
Memory optimization

- How to use the least memory possible
- Different kinds of optimization
- Depending on the number of values we want to stock
- We use a bitmap

Three type of representation

- Dense representation
- Sparse representation
- Delta varint encoding : use the sparse representation

Dense representation



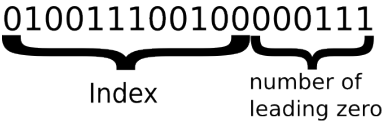
Pros

- Use the least possible amount of bits per value
- No index is stocked
- easy to access data
- Memory size of the bitmap constant

Cons

- When only few items are added, takes a lot of unnecessary space
- When checking for empty indexes, the whole bitmap needs to be read

Sparse representation



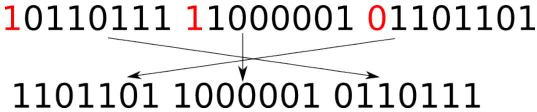
Pros

- Size of the map will fit the number of values we have

Cons

- It needs to stock the index AND the value
- Results in 20 bits for P = 14 and int 64

Delta varint encoder and decoder

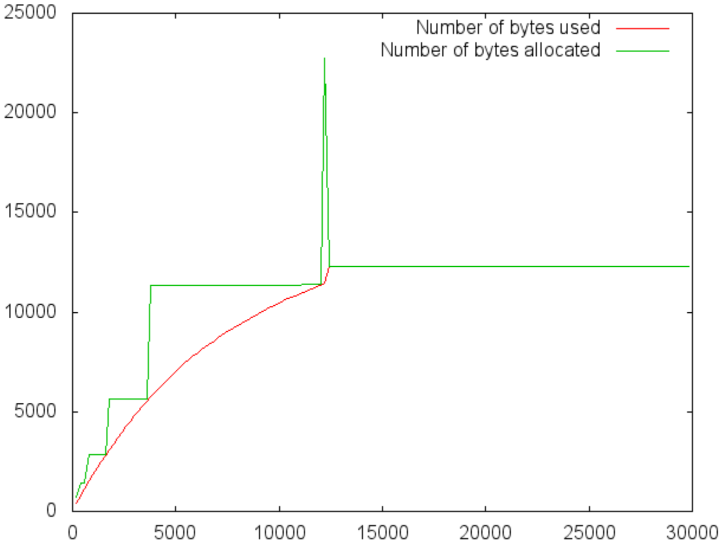


10110111 11000001 01101101

1101101 1000001 0110111

Principles

- Improves the sparse representation
- Will use the difference between current value and previous one
- It is used in order to decrease the sparse size



Number of bytes used and allocated by our bitmap in function of the number of addItem() calls

Conclusion

- We implemented it

Conclusion

- We implemented it
- Entirely (Except for P=25)

Conclusion

- We implemented it
- Entirely (Except for P=25)
- Using **C** (best language ever)

Conclusion

- We implemented it
- Entirely (Except for P=25)
- Using **C** (best language ever)
- **C99** would have been better

Conclusion

- We implemented it
- Entirely (Except for P=25)
- Using **C** (best language ever)
- **C99** would have been better
- It works !

Conclusion

- We implemented it
- Entirely (Except for P=25)
- Using **C** (best language ever)
- **C99** would have been better
- It works !

Bibliographie

- **P. Flajolet, Éric Fusy, O. Gandouet, and F. Meunier, HyperLogLog : the analysis of a near-optimalcardinality estimation algorithm**. In *In Analysis of Algorithms (AOFA)*, pages 127–146, 2007.

- **S. Heule, M. Nunkesser, A. Hall, HyperLogLog in Practice : Algorithmic Engineering of a State of The Art Cardinality Estimation Algorithm**.